WHAT IS CLAIMED:

1   1.   A method of performing native binding to execute native code during the

2   translation of subject program code executable by a subject processor to target program

3   code executable by a target processor, wherein native code is code executable by the

4   target processor, said method comprising:

5   identifying certain subject program code having corresponding native

6   code;

7   identifying the native code which corresponds to the identified subject

8   program code; and

9   executing the corresponding native code instead of executing a translated

10   version of the identified subject program code.

1   2.   The method of claim 1, wherein the identified subject program code

2   corresponds to a subject function and the identified native code corresponds to a native

3   function, wherein the native code executing step comprises:

4   executing the native function instead of the subject function in the

5   translation of the subject program code.

1   3.   The method of claim 2, wherein the native function executing step

2   comprises:

3   transforming zero or more function parameters from a target code

4   representation to a native code representation;

5   invoking the native function with the transformed function parameter

6   according to a prototype of the native function; and

7   transforming zero or more return values of the invoked native function

8   from a native code representation to a target code representation.

1   4.   The method of claim 3, wherein at least one of the transformations in the

2   transforming steps generates an intermediate representation of the transformation.

1   5.   The method of claim 3, wherein at least one of the transformations in the

2   transforming steps generates target code.

1   6.   The method of claim 3, wherein the native function executing step further

2   comprises:

3   transforming in target code all subject register values from the target code

4   representation to the native code representation;

5   invoking from target code a native code call stub function with the

6   transformed subject registers according to a uniform call stub interface; and

7   invoking from the native code call stub function the native function with

8   particular subject registers and/or parameter stack according to the prototype of

9   · the native function.

1   7.   The method of claim 3, wherein the native function executing step

2   comprises:

3   transforming a function parameter from a target code representation to a

4   native code representation;

5   invoking the native function with the transformed function parameter

6   according to a prototype of the native function; and

7   transforming a result of the invoked native function from a native code

8   representation to a target code representation.

1   8.   The method of claim 3, wherein the function parameter transforming step

2   and the native function invoking step are described in subject code by translator specific

3   instructions added to the subject instruction set.

1   9.   The method of claim 1, wherein the steps of identifying the certain subject

2   code and its corresponding native code are performed using a bind point description.

1   10.   The method of claim 9, wherein the bind point description includes a

2   subject function and a native function, wherein the subject function identifies the certain

3   subject program code having corresponding native code and the native function identifies

4   the corresponding native code.

1   11.   The method of claim 10, further comprising inserting in the target code a

2   call stub to the native function during translation of the subject code when encountering

3   the subject function contained in the bind point description.

1   12.   The method of claim 9, wherein the bind point description is embedded

2   within a translator performing the translation.

1   13.   The method of claim 9, further comprising reading the bind point

2   description from a stored bind point description file at the beginning of translation

3   execution.

1   14.   The method of claim 9, wherein the bind point description includes a

2   location in the subject code and a corresponding native function, wherein the location in

3   the subject code identifies the certain subject program code having corresponding native

4   code and the native function identifies the corresponding native code.

1   15.   The method of claim 9, wherein the bind point description includes a

2   location in the subject code and a reference to code to be invoked, wherein the location in

3   the subject code identifies the certain subject program code having corresponding native

4   code and the reference to code to be invoked identifies the corresponding native code.

1   16.   The method of claim 15, wherein the code to be invoked is target code.

1   17.   The method of claim 9, wherein the bind point description includes a

2   native function call which is inserted in the target code either before, after, or in place of

3   a subject function call.

1   18.   The method of claim 9, further performing runtime symbol patching

2   comprising:

3   encoding subject-to-native function mappings in a symbol table of the

4   subject program,

5   replacing entries in the symbol table of the subject program with special

6   native binding markers, and

7   interpreting the special native binding markers when encountered during

8   translation as bind point descriptions to identify an appropriate native function to

9   call.

1   19.   The method of claim 9, wherein the bind point description includes a

2   correspondence to an external Schizo call command, wherein the Schizo call command is

3   a translator-specific native binding instruction, the method comprising:

4   when encountering a bind point description identifying an external Schizo

5   call command during translation of the subject code, diverting the flow of

6   translation to the execution of the external Schizo call command.

1   20.   The method of claim 19, wherein the external Schizo call command

2   execution step comprises:

3   interpreting the external Schizo call command; and

4   generating an intermediate representation of the external Schizo call

5   command which:

6   transforms a function parameter from a target code representation

7   to a native code representation, and

8   invokes the native function with the transformed function

9   parameter according to a prototype of the native function.

1   21.   The method of claim 19, wherein the external Schizo call command

2   execution step comprises:

3   interpreting the external Schizo call command; and

4   generating target code for the external Schizo call command which:

5   transforms a function parameter from a target code representation

6   to a native code representation, and

7   invokes the native function with the transformed function

8   parameter according to a prototype of the native function.

1   22.   The method of claim 1, further comprising:

2   inserting Schizo call commands into the subject code, wherein Schizo call

3   commands are translator-specific native binding instructions; and

4   detecting the Schizo call commands during translation of the subject code.

1   23.   The method of claim 22, further comprising:

2   when encountering a Schizo call command during translation of the

3   subject code, diverting the flow of translation to the execution of the Schizo call

4   command.

1   24.   The method of claim 23, wherein the Schizo call command execution step

2   comprises:

3   interpreting the external Schizo call command; and

4   generating an intermediate representation of the Schizo call command

5   which:

6   transforms a function parameter from a target code representation to a

7   native code representation, and

8   invokes the native function with the transformed function parameter

9   according to a prototype of the native function.

1   25.   The method of claim 23, wherein the Schizo call command execution step

2 comprises:

3   interpreting the Schizo call command; and

4   generating target code for the Schizo call command which:

5   transforms a function parameter from a target code representation

6   to a native code representation, and

7   invokes the native function with the transformed function

8   parameter according to a prototype of the native function

1   26.   The method of claim 22, wherein the Schizo call commands are variable

2 length instructions including multiple sub-component instructions.

1   27.   The method of claim 26, wherein the multiple sub-component instructions

2 include a Schizo Escape sub-component instruction, said Schizo call commands detecting

3 step further comprising detecting the Schizo Escape sub-component instruction.

1   28.   The method of claim 27, wherein said Schizo Escape sub-component

2 instruction further identifies a type of Schizo call command represented by the other sub-

3 component instructions of the Schizo call command.

1   29.   The method of claim 1, further comprising:

2   parsing and decoding a native binding implementation scripting language

3 containing native binding scripts;

4   interpreting the native binding scripts during translation;

5   generating an intermediate representation of the native binding scripts to

6   transform a function parameter from a target code representation to a native code

7   representation.

1   30.   The method of claim 29, further comprising:

2   integrating the intermediate representation of the native binding scripts

3   into an intermediate representation forest for a block of subject code; and

4   generating target code for the intermediate representation forest.

1   31.   The method of claim 1, further comprising:

2   transforming in target code all subject register values from the target code

3   representation to the native code representation;

4   invoking from target code a native code call stub function with the

5   transformed subject registers according to a uniform call stub interface;

6   interpreting the native code call stub function; and

7   generating an intermediate representation of the native code call stub

8   function binding scripts to transform a function parameter from a target code

9   representation to a native code representation.

1   32.   The method of claim 21, further comprising:

2   integrating the intermediate representation of the native code call stub

3   function into an intermediate representation forest for a block of subject code; and

4   generating target code for the intermediate representation forest.

1   33.   The method of claim 3, wherein the native function executing step further

2   comprises:

3   transforming in target code all subject register values from the target code

4   representation to the native code representation;

5   invoking from target code a native code call stub function with the

6   transformed subject registers; and

7   invoking from the native code call stub function the native function with

8   particular subject registers and/or parameter stack according to the prototype of

9   the native function.

1   34.   The method of claim 1, further comprising:

2   parsing a scripting language implementation of a native code call stub

3   function;

4   compiling the parsed native code call stub function into a native code

5   executable module; and

6   linking the native code executable module with an executable for

7   performing the translation.

1   35.   The method of claim 34, wherein the native code executable module is

2   executable for:

3   transforming in target code all subject register values from the target code

4   representation to the native code representation;

5   invoking from target code a native code call stub function with the

6   transformed subject registers; and

7 invoking from the native code call stub function the native function with

8 particular subject registers and/or parameter stack according to the prototype of

9 the native function.

1 36. The method of claim 34, wherein the steps of identifying the certain

2 subject code and its corresponding native code are performed using a bind point

3 description, said bind point description including a subject function and a native code call

4 stub function, wherein the subject function identifies the certain subject program code

5 having corresponding native code and the native code call stub function identifies the

6 corresponding native code.

1 37. The method of claim 36, further comprising encoding the identity of the

2 native function of the native code call stub function in the scripting language

3 implementation of the native code executable module.

1 38. The method of claim 3, wherein the native function executing step further

2 comprises:

3 transforming in target code all subject register values from the target code

4 representation to the native code representation;

5 invoking from target code a target code call stub function with the

6 transformed subject registers; and

7 invoking from the target code call stub function the native function with

8 particular subject registers and/or parameter stack according to the prototype of

9 the native function.

1   39.   The method of claim 38, further comprising:

2   generating an intermediate representation of the native function executing

3   step;

4   integrating the intermediate representation of the native function executing

5   step into an intermediate representation forest for a block of subject code; and

6   generating target code for the intermediate representation forest.

1   40.   The method of claim 1, wherein the subject function to be executed is a

2   system call.

1   41.   The method of claim 1, wherein the subject function to be executed is a

2   library function.

1   42.   A computer-readable storage medium having software resident thereon in

2   the form of computer-readable code executable by a computer to perform the following

3   native binding steps to execute native code during the translation of subject program code

4   executable by a subject processor to target program code executable by a target

5   processor, wherein native code is code executable by the target processor, said steps

6   comprising:

7   identifying certain subject program code having corresponding native

8   code;

9   identifying the native code which corresponds to the identified subject

10   program code; and

11   executing the corresponding native code instead of executing a translated

12   version of the identified subject program code.

1   43.   The computer-readable storage medium of claim 42, wherein the

2   identified subject program code corresponds to a subject function and the identified

3   native code corresponds to a native function, wherein the native code executing step

4   comprises:

5   executing the native function instead of the subject function in the

6   translation of the subject program code.

1   44.   The computer-readable storage medium of claim 43, wherein the native

2   function executing step comprises:

3   transforming zero or more function parameters from a target code

4   representation to a native code representation;

5   invoking the native function with the transformed function parameter

6   according to a prototype of the native function; and

7   transforming zero or more return values of the invoked native function

8   from a native code representation to a target code representation.

1   45.   The computer-readable storage medium of claim 44, wherein at least one

2   of the transformations in the transforming steps generates an intermediate representation

3   of the transformation.

1   46.   The computer-readable storage medium of claim 44, wherein at least one

2   of the transformations in the transforming steps generates target code.

1   47.   The computer-readable storage medium of claim 44, wherein the native

2   function executing step further comprises:

3   transforming in target code all subject register values from the target code

4   representation to the native code representation;

5   invoking from target code a native code call stub function with the

6   transformed subject registers according to a uniform call stub interface; and

7   invoking from the native code call stub function the native function with

8   particular subject registers and/or parameter stack according to the prototype of

9   the native function.

1   48.   The computer-readable storage medium of claim 44, wherein the native

2   function executing step comprises:

3   transforming a function parameter from a target code representation to a

4   native code representation;

5   invoking the native function with the transformed function parameter

6   according to a prototype of the native function; and

7   transforming a result of the invoked native function from a native code

8   representation to a target code representation.

1   49.   The computer-readable storage medium of claim 44, wherein the function

2   parameter transforming step and the native function invoking step are described in

3   subject code by translator specific instructions added to the subject instruction set.

1   50.   The computer-readable storage medium of claim 42, wherein the steps of

2   identifying the certain subject code and its corresponding native code are performed

3   using a bind point description.

1   51.   The computer-readable storage medium of claim 50, wherein the bind

2   point description includes a subject function and a native function, wherein the subject

3   function identifies the certain subject program code having corresponding native code

4   and the native function identifies the corresponding native code.

1   52.   The computer-readable storage medium of claim 51, said computer-

2   readable code executable further executable for inserting in the target code a call stub to

3   the native function during translation of the subject code when encountering the subject

4   function contained in the bind point description.

1   53.   The computer-readable storage medium of claim 50, wherein the bind

2   point description is embedded within a translator performing the translation.

1   54.   The computer-readable storage medium of claim 50, said computer-

2   readable code executable further executable for reading the bind point description from a

3   stored bind point description file at the beginning of translation execution.

1   55.   The computer-readable storage medium of claim 50, wherein the bind

2   point description includes a location in the subject code and a corresponding native

3   function, wherein the location in the subject code identifies the certain subject program

4   code having corresponding native code and the native function identifies the

5   corresponding native code.

1   56.   The computer-readable storage medium of claim 50, wherein the bind

2   point description includes a location in the subject code and a reference to code to be

3   invoked, wherein the location in the subject code identifies the certain subject program

4   code having corresponding native code and the reference to code to be invoked identifies

5   the corresponding native code.

1   57.   The computer-readable storage medium of claim 56, wherein the code to

2   be invoked is target code.

1   58.   The computer-readable storage medium of claim 50, wherein the bind

2   point description includes a native function call which is inserted in the target code either

3   before, after, or in place of a subject function call.

1   59.   The computer-readable storage medium of claim 50, said computer-

2   readable code executable further executable for performing runtime symbol patching

3   comprising:

4   encoding subject-to-native function mappings in a symbol table of the

5   subject program,

6   replacing entries in the symbol table of the subject program with special

7   native binding markers, and

8   interpreting the special native binding markers when encountered during

9   translation as bind point descriptions to identify an appropriate native function to

10   call.

1   60.   The computer-readable storage medium of claim 50, wherein the bind

2 point description includes a correspondence to an external Schizo call command, wherein

3 the Schizo call command is a translator-specific native binding instruction, said

4 computer-readable code executable further executable for:

5   when encountering a bind point description identifying an external Schizo

6   call command during translation of the subject code, diverting the flow of

7   translation to the execution of the external Schizo call command.

1   61.   The computer-readable storage medium of claim 60, wherein the external

2 Schizo call command execution step comprises:

3   interpreting the external Schizo call command; and

4   generating an intermediate representation of the external Schizo call

5   command which:

6   transforms a function parameter from a target code representation

7   to a native code representation, and

8   invokes the native function with the transformed function

9   parameter according to a prototype of the native function.

1   62.   The computer-readable storage medium of claim 60, wherein the external

2 Schizo call command execution step comprises:

3   interpreting the external Schizo call command; and

4   generating target code for the external Schizo call command which:

5   transforms a function parameter from a target code representation

6   to a native code representation, and

7   invokes the native function with the transformed function

8   parameter according to a prototype of the native function.

1   63.   The computer-readable storage medium of claim 42, said computer-

2   readable code executable further executable for performing the following steps:

3   inserting Schizo call commands into the subject code, wherein Schizo call

4   commands are translator-specific native binding instructions; and

5   detecting the Schizo call commands during translation of the subject code.

1   64.   The computer-readable storage medium of claim 63, said computer-

2   readable code executable further executable for performing the following steps:

3   when encountering a Schizo call command during translation of the

4   subject code, diverting the flow of translation to the execution of the Schizo call

5   command.

1   65.   The computer-readable storage medium of claim 64, wherein the Schizo

2   call command execution step comprises:

3   interpreting the external Schizo call command; and

4   generating an intermediate representation of the Schizo call command

5   which:

6   transforms a function parameter from a target code representation

7   to a native code representation, and

8   invokes the native function with the transformed function

9   parameter according to a prototype of the native function.

1   66.   The computer-readable storage medium of claim 64, wherein the Schizo

2   call command execution step comprises:

3   interpreting the Schizo call command; and

4   generating target code for the Schizo call command which:

5   transforms a function parameter from a target code representation

6   to a native code representation, and

7   invokes the native function with the transformed function

8   parameter according to a prototype of the native function

1   67.   The computer-readable storage medium of claim 63, wherein the Schizo

2   call commands are variable length instructions including multiple sub-component

3   instructions.

1   68.   The computer-readable storage medium of claim 67, wherein the multiple

2   sub-component instructions include a Schizo Escape sub-component instruction, said

3   Schizo call commands detecting step further comprising detecting the Schizo Escape sub-

4   component instruction.

1   69.   The computer-readable storage medium of claim 68, wherein said Schizo

2   Escape sub-component instruction further identifies a type of Schizo call command

3   represented by the other sub-component instructions of the Schizo call command.

1   70.   The computer-readable storage medium of claim 42, said computer-

2 readable code executable further executable for performing the following steps:

3   parsing and decoding a native binding implementation scripting language

4 containing native binding scripts;

5   interpreting the native binding scripts during translation; and

6   generating an intermediate representation of the native binding scripts to

7 transform a function parameter from a target code representation to a native code

8 representation.

1   71.   The computer-readable storage medium of claim 70, said computer-

2 readable code executable further executable for performing the following steps:

3   integrating the intermediate representation of the native binding scripts

4 into an intermediate representation forest for a block of subject code; and

5   generating target code for the intermediate representation forest.

1   72.   The computer-readable storage medium of claim 42, said computer-

2 readable code executable further executable for performing the following steps:

3   transforming in target code all subject register values from the target code

4 representation to the native code representation;

5   invoking from target code a native code call stub function with the

6 transformed subject registers according to a uniform call stub interface;

7   interpreting the native code call stub function; and

8   generating an intermediate representation of the native code call stub

9   function binding scripts to transform a function parameter from a target code

10   representation to a native code representation.

1   73.   The computer-readable storage medium of claim 62, said computer-

2   readable code executable further executable for performing the following steps:

3   integrating the intermediate representation of the native code call stub

4   function into an intermediate representation forest for a block of subject code; and

5   generating target code for the intermediate representation forest

1   74.   The computer-readable storage medium of claim 44, wherein the native

2   function executing step further comprises:

3   transforming in target code all subject register values from the target code

4   representation to the native code representation;

5   invoking from target code a native code call stub function with the

6   transformed subject registers; and

7   invoking from the native code call stub function the native function with

8   particular subject registers and/or parameter stack according to the prototype of

9   the native function.

1   75.   The computer-readable storage medium of claim 42, said computer-

2   readable code executable further executable for performing the following steps:

3   parsing a scripting language implementation of a native code call stub

4   function;

5   compiling the parsed native code call stub function into a native code

6   executable module; and

7   linking the native code executable module with an executable for

8   performing the translation.

1   76.   The computer-readable storage medium of claim 75, wherein the native

2   code executable module is executable for:

3   transforming in target code all subject register values from the target code

4   representation to the native code representation;

5   invoking from target code a native code call stub function with the

6   transformed subject registers; and

7   invoking from the native code call stub function the native function with

8   particular subject registers and/or parameter stack according to the prototype of

9   the native function.

1   77.   The computer-readable storage medium of claim 75, wherein the steps of

2   identifying the certain subject code and its corresponding native code are performed

3   using a bind point description, said bind point description including a subject function

4   and a native code call stub function, wherein the subject function identifies the certain

5   subject program code having corresponding native code and the native code call stub

6   function identifies the corresponding native code.

1   78.   The computer-readable storage medium of claim 77, said computer-

2   readable code executable further executable for encoding the identity of the native

3   function of the native code call stub function in the scripting language implementation of

4   the native code executable module.

1   79.   The computer-readable storage medium of claim 44, wherein the native

2   function executing step further comprises:

3   transforming in target code all subject register values from the target code

4   representation to the native code representation;

5   invoking from target code a target code call stub function with the

6   transformed subject registers; and

7   invoking from the target code call stub function the native function with

8   particular subject registers and/or parameter stack according to the prototype of

9   the native function.

1   80.   The computer-readable storage medium of claim 79, said computer-

2   readable code executable further executable for performing the following steps:

3   generating an intermediate representation of the native function executing

4   step;

5   integrating the intermediate representation of the native function executing

6   step into an intermediate representation forest for a block of subject code; and

7   generating target code for the intermediate representation forest.

1   81.   The computer-readable storage medium of claim 42, wherein the subject

2   function to be executed is a system call.

1   82.   The computer-readable storage medium of claim 42, wherein the subject

2   function to be executed is a library function.

1   83.   In combination:

2   a target processor; and

3   translator code for performing native binding to execute native code

4   during the translation of subject program code executable by a subject processor

5   to target program code executable by a target processor, wherein native code is

6   code executable by the target processor, said translator code comprising code

7   executable by said target processor for performing the following steps:

8   identifying certain subject program code having corresponding

9   native code;

10   identifying the native code which corresponds to the identified

11   subject program code; and

12   executing the corresponding native code instead of executing a

13   translated version of the identified subject program code.

1   84.   The combination of claim 83, wherein the identified subject program code

2   corresponds to a subject function and the identified native code corresponds to a native

3   function, wherein the native code executing step comprises:

4   executing the native function instead of the subject function in the

5   translation of the subject program code.

1   85.   The combination of claim 84, wherein the native function executing step

2   comprises:

3   transforming zero or more function parameters from a target code

4   representation to a native code representation;

5   invoking the native function with the transformed function parameter

6   according to a prototype of the native function; and

7   transforming zero or more return values of the invoked native function

8   from a native code representation to a target code representation.

1   86.   The combination of claim 85, wherein at least one of the transformations

2   in the transforming steps generates an intermediate representation of the transformation.

1   87.   The combination of claim 85, wherein at least one of the transformations

2   in the transforming steps generates target code.

1   88.   The combination of claim 85, wherein the native function executing step

2   further comprises:

3   transforming in target code all subject register values from the target code

4   representation to the native code representation;

5   invoking from target code a native code call stub function with the

6   transformed subject registers according to a uniform call stub interface; and

7   invoking from the native code call stub function the native function with

8   particular subject registers and/or parameter stack according to the prototype of

9   the native function.

1   89.   The combination of claim 85, wherein the native function executing step

2   comprises:

3   transforming a function parameter from a target code representation to a

4   native code representation;

5   invoking the native function with the transformed function parameter

6   according to a prototype of the native function; and

7   transforming a result of the invoked native function from a native code

8   representation to a target code representation.

1   90.   The combination of claim 85, wherein the function parameter

2   transforming step and the native function invoking step are described in subject code by

3   translator specific instructions added to the subject instruction set.

1   91.   The combination of claim 83, wherein the steps of identifying the certain

2   subject code and its corresponding native code are performed using a bind point

3   description.

1   92.   The combination of claim 91, wherein the bind point description includes

2   a subject function and a native function, wherein the subject function identifies the

3   certain subject program code having corresponding native code and the native function

4   identifies the corresponding native code.

1   93.   The combination of claim 92, said translator code further comprising code

2   executable by said target processor for inserting in the target code a call stub to the native

3 function during translation of the subject code when encountering the subject function

4 contained in the bind point description.

1 94. The combination of claim 91, wherein the bind point description is

2 embedded within a translator performing the translation.

1 95. The combination of claim 91, said translator code further comprising code

2 executable by said target processor for reading the bind point description from a stored

3 bind point description file at the beginning of translation execution.

1 96. The combination of claim 91, wherein the bind point description includes

2 a location in the subject code and a corresponding native function, wherein the location in

3 the subject code identifies the certain subject program code having corresponding native

4 code and the native function identifies the corresponding native code.

1 97. The combination of claim 91, wherein the bind point description includes

2 a location in the subject code and a reference to code to be invoked, wherein the location

3 in the subject code identifies the certain subject program code having corresponding

4 native code and the reference to code to be invoked identifies the corresponding native

5 code.

1 98. The combination of claim 97, wherein the code to be invoked is target

2 code.

1   99.   The combination of claim 91, wherein the bind point description includes

2   a native function call which is inserted in the target code either before, after, or in place

3   of a subject function call.

1   100.   The combination of claim 91, said translator code further comprising code

2   executable by said target processor for performing runtime symbol patching comprising:

3   encoding subject-to-native function mappings in a symbol table of the

4   subject program,

5   replacing entries in the symbol table of the subject program with special

6   native binding markers, and

7   interpreting the special native binding markers when encountered during

8   translation as bind point descriptions to identify an appropriate native function to

9   call.

1   101.   The combination of claim 91, wherein the bind point description includes

2   a correspondence to an external Schizo call command, wherein the Schizo call command

3   is a translator-specific native binding instruction, the method comprising:

4   when encountering a bind point description identifying an external Schizo

5   call command during translation of the subject code, diverting the flow of

6   translation to the execution of the external Schizo call command.

1   102.   The combination of claim 101, wherein the external Schizo call command

2   execution step comprises:

3   interpreting the external Schizo call command; and

4   generating an intermediate representation of the external Schizo call

5   command which:

6   transforms a function parameter from a target code representation

7   to a native code representation, and

8   invokes the native function with the transformed function

9   parameter according to a prototype of the native function.

1   103.   The combination of claim 101, wherein the external Schizo call command

2   execution step comprises:

3   interpreting the external Schizo call command; and

4   generating target code for the external Schizo call command which:

5   transforms a function parameter from a target code representation

6   to a native code representation, and

7   invokes the native function with the transformed function

8   parameter according to a prototype of the native function.

1   104.   The combination of claim 83, said translator code further comprising code

2   executable by said target processor for performing the following steps:

3   inserting Schizo call commands into the subject code, wherein Schizo call

4   commands are translator-specific native binding instructions; and

5   detecting the Schizo call commands during translation of the subject code.

1   105.   The combination of claim 104, said translator code further comprising

2   code executable by said target processor for performing the following steps:

3   when encountering a Schizo call command during translation of the

4   subject code, diverting the flow of translation to the execution of the Schizo call

5   command.

1   106.   The combination of claim 105, wherein the Schizo call command

2   execution step comprises:

3   interpreting the external Schizo call command; and

4   generating an intermediate representation of the Schizo call command

5   which:

6   transforms a function parameter from a target code representation

7   to a native code representation, and

8   invokes the native function with the transformed function

9   parameter according to a prototype of the native function.

1   107.   The combination of claim 105, wherein the Schizo call command

2   execution step comprises:

3   interpreting the Schizo call command; and

4   generating target code for the Schizo call command which:

5   transforms a function parameter from a target code representation

6   to a native code representation, and

7   invokes the native function with the transformed function

8   parameter according to a prototype of the native function.

1   108.   The combination of claim 104, wherein the Schizo call commands are

2   variable length instructions including multiple sub-component instructions.

1   109.   The combination of claim 108, wherein the multiple sub-component

2   instructions include a Schizo Escape sub-component instruction, said Schizo call

3   commands detecting step further comprising detecting the Schizo Escape sub-component

4   instruction.

1   110.   The combination of claim 109, wherein said Schizo Escape sub-

2   component instruction further identifies a type of Schizo call command represented by

3   the other sub-component instructions of the Schizo call command.

1   111.   The combination of claim 83, said translator code further comprising code

2   executable by said target processor for performing the following steps:

3   parsing and decoding a native binding implementation scripting language

4   containing native binding scripts;

5   interpreting the native binding scripts during translation; and

6   generating an intermediate representation of the native binding scripts to

7   transform a function parameter from a target code representation to a native code

8   representation.

1   112.   The combination of claim 111, said translator code further comprising

2   code executable by said target processor for performing the following steps:

3   integrating the intermediate representation of the native binding scripts

4   into an intermediate representation forest for a block of subject code; and

5   generating target code for the intermediate representation forest.

1   113.   The combination of claim 83, said translator code further comprising code

2   executable by said target processor for performing the following steps:

3   transforming in target code all subject register values from the target code

4   representation to the native code representation;

5   invoking from target code a native code call stub function with the

6   transformed subject registers according to a uniform call stub interface;

7   interpreting the native code call stub function; and

8   generating an intermediate representation of the native code call stub

9   function binding scripts to transform a function parameter from a target code

10   representation to a native code representation.

1   114.   The combination of claim 103, said translator code further comprising

2   code executable by said target processor for performing the following steps:

3   integrating the intermediate representation of the native code call stub

4   function into an intermediate representation forest for a block of subject code; and

5   generating target code for the intermediate representation forest.

1   115.   The combination of claim 85, wherein the native function executing step

2   further comprises:

3   transforming in target code all subject register values from the target code

4   representation to the native code representation;

5   invoking from target code a native code call stub function with the

6   transformed subject registers;

7   invoking from the native code call stub function the native function with

8   particular subject registers and/or parameter stack according to the prototype of

9   the native function.

1   116.   The combination of claim 83, said translator code further comprising code

2 executable by said target processor for performing the following steps:

3   parsing a scripting language implementation of a native code call stub

4   function;

5   compiling the parsed native code call stub function into a native code

6   executable module; and

7   linking the native code executable module with an executable for

8   performing the translation.

1   117.   The combination of claim 116, wherein the native code executable module

2 is executable for:

3   transforming in target code all subject register values from the target code

4   representation to the native code representation;

5   invoking from target code a native code call stub function with the

6   transformed subject registers; and

7   invoking from the native code call stub function the native function with

8   particular subject registers and/or parameter stack according to the prototype of

9   the native function.

1   118.   The combination of claim 116, wherein the steps of identifying the certain

2 subject code and its corresponding native code are performed using a bind point

3   description, said bind point description including a subject function and a native code call

4   stub function, wherein the subject function identifies the certain subject program code

5   having corresponding native code and the native code call stub function identifies the

6   corresponding native code.

1   119.   The combination of claim 118, said translator code further comprising

2   code executable by said target processor for encoding the identity of the native function

3   of the native code call stub function in the scripting language implementation of the

4   native code executable module.

1   120.   The combination of claim 85, wherein the native function executing step

2   further comprises:

3   transforming in target code all subject register values from the target code

4   representation to the native code representation;

5   invoking from target code a target code call stub function with the

6   transformed subject registers; and

7   invoking from the target code call stub function the native function with

8   particular subject registers and/or parameter stack according to the prototype of

9   the native function.

1   · 121.   The combination of claim 120, said translator code further comprising

2   code executable by said target processor for performing the following steps:

3   generating an intermediate representation of the native function executing

4   step;

5   integrating the intermediate representation of the native function executing

6   step into an intermediate representation forest for a block of subject code; and

7   generating target code for the intermediate representation forest.

1   122.   The combination of claim 83, wherein the subject function to be executed

2 is a system call.

1   123.   The combination of claim 83, wherein the subject function to be executed

2 is a library function.